Alineamiento de múltiples secuencias de aminoácidos usando algoritmos genéticos

Luis Felipe Solanilla
solanilla@caliescali.com

Carlos Arturo Gómez Teshima
Teshima82@ yahoo.com

Luis Eduardo Múnera
lemunera@ icesi.edu.co

Fecha de recepción: 15-12-2004
Fecha de aceptación: 01-04-2005

ABSTRACT

This article describes a genetic algorithm and its implementation that it allows the alignment of multiple sequences of amino acids by means of relocations that simulate the gaps insertion (holes) and actions of recombination to obtain higher scores in the alignment. Such scores are obtained by means of the method of the sum of even, in which there are considered all the possible combinations of amino acids in each column of the alignment and they are qualified being based on the punctuations of the matrix BLOSUM62.

KEY WORDS

Bioinformatics, Genetic Algorithms, Genomics, Multiple Sequence Alignment.

RESUMEN

El presente artículo describe un algoritmo genético y su implementación, que permite el alineamiento de múltiples secuencias de aminoácidos mediante reacomodaciones que simulan la inserción de gaps (huecos) y acciones de recombinación para obtener puntajes más altos en el alineamiento. Tales puntajes se obtienen mediante el método de la suma de pares, en el cual se consideran todas las posibles combinaciones de aminoácidos en cada columna del alineamiento y se califican basándose en las puntuaciones de la matriz BLOSUM62.

PALABRAS CLAVE

Bioinformática, Algoritmos Genéticos, Genómica, Alineamiento de Múltiples Secuencias.

Clasificación Colciencias: A

1. INTRODUCCIÓN

Desde el principio de los años noventa muchas entidades gubernamentales y privadas han analizado el genoma de varias especies, tales como levaduras, bacterias, ratones y otros seres (incluyendo el humano). Durante estos esfuerzos de colaboración se han generado cantidades de información que se recogen y se almacenan en grandes bases de datos, la mayoría de las cuales son públicas y accesibles desde cualquier parte del mundo.

De la misma forma, cuando se va a introducir una nueva cadena de aminoácidos, se debe comparar primero la secuencia de componentes con todas las existentes, para poder clasificarla. Sin embargo, el manejo de tal tipo de clasificación no se puede hacer a mano debido a la gran cantidad de información que se manipula y las complicadas formas de comparación y asociación que se han creado a medida que pasa el tiempo. Por tal motivo se han creado diversos programas que facilitan esta tarea.

Una de las importantes contribuciones de la biología molecular al análisis evolutivo es el descubrimiento de que las secuencias de ADN de diferentes organismos se encuentran a menudo relacionadas. Aquí, genes similares son conservados a través de diferentes especies divergentes, a menudo desempeñando una función similar o incluso idéntica, y en otras ocasiones queda reacomodándose para desempeñar una función alterada a través de las fuerzas de la selección natural. A través de alineamiento múltiple de estas secuencias, los patrones de secuencia que han sido sujeto de alteración pueden ser analizados.

El alineamiento de múltiples secuencias (AMS) de un conjunto de secuencias puede también ser visto como la historia evolutiva de las secuencias. Así, si las secuencias en el AMS se alinean muy bien, parecería que han sido recientemente derivadas de un ancestro común. En contraste, un grupo de secuencias pobremente alineadas comparten un relación evolutiva distante y compleja. La tarea de alinear un conjunto de secuencias, algunas más relacionadas que otras, es idéntica a descubrir las relaciones evolutivas entre las secuencias.

Al igual que con el alineamiento de pares de secuencias, la dificultad en alinear un grupo de secuencias varía considerablemente con la similaridad de las secuencias. Por un lado, si la cantidad de variaciones en las secuencias es mínima, es relativamente sencillo alinearlas, aun sin la asistencia de un programa de computador. Por otro lado, si la cantidad de variaciones es grande, podría ser muy difícil encontrar un alineamiento óptimo de las secuencias porque muchas combinaciones de sustituciones, inserciones y eliminaciones, cada una prediciendo un alineamiento diferente, son posibles.

Para la construcción de los AMS existen varios enfoques. El primero, en el que se busca encontrar el alineamiento óptimo agotando todas las posibilidades existentes, pero si se tiene en cuenta el número de posibilidades del que se habla se puede intuir que no será aplicable a un número relativamente grande de cadenas. Es así como este enfoque se usa como máximo en el alineamiento de seis secuencias, las que a su vez deben ser relativamente cortas. El se-

gundo es el de los métodos progresivos en los que se aplica el primer enfoque a las secuencias más relacionadas y posteriormente se van agregando poco a poco secuencias menos relacionadas al alineamiento. Dentro de este segundo enfoque encontramos importantes representantes como Clustalw y Pileup. Este segundo enfoque tiene un problema importante y es la dependencia del resultado de los métodos en los primeros AMS de las primeras secuencias junto con el hecho de que a medida que se agregan las secuencias menos relacionadas, se pasa de un alineamiento parcial con muy buenos parciales a un AMS en el que en cada inserción provoca una propagación de errores en todo el alineamiento. Un tercer enfoque es el de los métodos iterativos en el que lo que se busca es lograr mejorar poco a poco el puntaje general del alineamiento. Esto se logra realineando repetidamente subgrupos de las secuencias y luego alineando esos subgrupos en un alineamiento global con todas las secuencias. La selección de los subgrupos se puede realizar separando una o dos secuencias del resto, realizando un estudio del árbol filogenético o ejecutando una selección aleatoria. Dentro de este enfoque se encuentran algoritmos como MultAlin, DIALIGN, HMM (Hidden Markov Models) y los Algoritmos Genéticos, que son los que acaparan el interés de este trabajo.[1]

2. CONCEPTOS BÁSICOS

2.1 Nucleótido

Compuesto químico formado por la unión de una molécula de ácido fosfórico, un azúcar de cinco átomos de carbono y una base nitrogenada derivada de la purina o la pirimidina. Los nucleótidos son las unidades constituyentes de los ácidos nucleicos. También se encuentran libres en las células y forman parte de ciertas coenzimas. La Tabla 1 muestra el código usado para expresar las bases nucleótidas.[3]

Tabla 1. Código de bases nucleótidas.

Símbolo	Significado	Explicación
G	G	Guanina
A	A	Adenina
T	T	Tiamina
C	C	Citosina
R	A o G	Purina
Y	C o T	Pirimidina
M	A o C	Amino
K	G o T	Keto
S	C o G	Interacción fuerte
W	A o T	Interacción débil
H	A, C o T	H sigue a G en el alfabeto
B	C, G o T	B sigue a A en el alfabeto
V	A, C o G no T (no U)	V sigue a U en el alfabeto
D	A, G o T no C	D sigue a C en el alfabeto
N	A, C, G o T	Cualquier base

2.2 Aminoácidos

Compuestos orgánicos que contienen un grupo amino ($8NH_2$) y un grupo carboxilo (8COOH). Veinte de estos compuestos son los constituyentes de las proteínas. La Tabla 2 presenta el código estándar usado para representar los aminoácidos.[3]

2.3 Alineamiento de secuencias

Comparación lineal de secuencias aminoacídicas (o ácidos nucleicos) en la que se introducen inserciones para hacer que posiciones equivalentes en secuencias adyacentes se sitúen en el registro correcto. Los alineamientos son la base de los métodos de análisis de secuencias. La Figura 1 muestra un ejemplo de lo que es un alineamiento con Gaps (Huecos).[2]

Existen dos tipos de alineamientos:

- **Global:** El alineamiento global son las posibles coincidencias existentes a lo largo de toda la secuencia del aminoácido o nucleótido. Tratando siempre de encontrar el mayor número de coincidencias posibles (ver Figura 2).
- **Local:** Un alineamiento local se hace en pequeñas fracciones de la cadena original en donde existen regiones idénticas o de alta similaridad. La prioridad dentro de este tipo de alineamiento es encontrar esas regiones locales antes que encontrar coincidencias entre cadenas vecinas o pares de aminoácidos (ver Figura 3).

Tabla 2. Código estándar de aminoácidos.

Código de 1 letra	Código de letras	Aminoácido
A	Ala	Alanina
C	Cys	Cysteína
D	Asp	Ácido aspártico
E	Glu	Ácido glutámico
F	Phe	Phenylalanina
G	Gly	Glicina
H	His	Histidina
I	Ile	Isoleucina
K	Lys	Lysina
L	Leu	Leucina
M	Met	Methionina
N	Asn	Asparagina
P	Pro	Prolina
Q	Gln	Glutamina
R	Arg	Arginina
S	Ser	Serina
T	Thr	Threonina
V	Val	Valina
W	Trp	Tryptophan
X	Xxx	Aminoácido no determinado
Y	Tyr	Tyrosina
Z	Glx	Glutamina u otro glutámico

Secuencia A	A	G	▲	▲	C	D	E	V	I	G
Secuencia B	A	G	E	Y	C	D	▲	I	I	G

Figura 1. Ejemplo de alineamiento con GAPS.

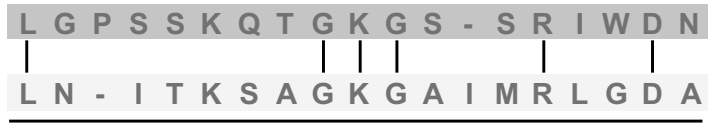

Figura 2. Ejemplo de un alineamiento global.

Figura 3. Ejemplo de un alineamiento local.

2.4 Puntuación del alineamiento

Los valores matriciales (Tabla 3) se basan en las sustituciones de aminoácido observadas en un gran conjunto de aproximadamente 2.000 bloques (patrones de aminoácido). Estos bloques fueron localizados en una base de datos de secuencias de proteínas que representaban más de 500 familias de proteínas relacionadas y que actuaban como identificadores de esas familias.[1]

MATRIZ BLOSUM62

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A Ala	4																			
R Arg	-1	5																		
N Asn	-2	0	6																	
D Asp	-2	-2	1	6																
C Cys	0	-3	-3	-3	9															
Q Gln	-1	1	0	0	-3	5														
E Glu	-1	0	0	2	-4	2	5													
G Gly	0	-2	0	-1	-3	-2	-2	6												
H His	-2	0	1	-1	-3	0	0	-2	8											
I Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
L Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
K Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
M Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
F Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
P Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
S Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
T Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
W Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Y Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
V Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

Tabla 3. Matriz Blosum62.

2.5. Algoritmos genéticos

Los algoritmos genéticos usan una analogía directa con el comportamiento natural. Trabajan con una población de individuos, cada uno de los cuales representa una solución factible a un problema dado. A cada individuo se le asigna un valor o puntuación (*fitness*), relacionado con la bondad de dicha solución. En la naturaleza esto equivaldría al grado de efectividad de un organismo para competir por unos determinados recursos. Cuanto mayor sea la adaptación de un individuo al problema, mayor será la probabilidad de que el mismo sea seleccionado para reproducirse, cruzando (*crossover*) su material genético con otro individuo seleccionado de igual forma. Este cruce producirá nuevos individuos, descendientes de los anteriores, los cuales comparten algunas de las características de sus padres. Cuanto menor sea la adaptación de un individuo menor será la probabilidad de que sea seleccionado para la reproducción, y por tanto de que su material genético se propague en sucesivas generaciones. De esta manera se produce una nueva población de posibles soluciones, la cual reemplaza a la anterior y verifica la interesante propiedad de que contiene una mayor proporción de buenas características en comparación con la población anterior. Así, a lo largo de las generaciones las buenas características se propagan a través de la población. Favoreciendo el cruce de los individuos mejor adaptados, se van explorando las áreas más prometedoras del espacio de búsqueda. Si el algoritmo genético ha sido bien diseñado, la población convergerá hacia una solución óptima del problema. Partiendo de lo anterior, un algoritmo genético consiste en lo siguiente: hallar de qué parámetros depende el problema, codificarlos en un cromosoma, definir la función de *fitness* y posteriormente aplicar los métodos de la evolución: selección y reproducción sexual con intercambio de información y alteraciones que generan diversidad. Los algoritmos genéticos están inspirados en la naturaleza, en la evolución de las especies.[4]

Los algoritmos genéticos requieren que el conjunto se codifique en un *cromosoma.* Cada cromosoma tiene varios genes, que corresponden a sendos parámetros del problema. Para poder trabajar con estos genes en el ordenador es necesario codificarlos en una *cadena,* es decir, una secuencia de símbolos (números o letras) que generalmente va a estar compuesta de 0s y 1s. Tras haber definido la codificación de las variables el algoritmo genético procede de la siguiente forma:

1. Evaluar la puntuación (*fitness*) de cada uno de los individuos.
2. Permitir a cada uno de los individuos reproducirse según su puntuación.
3. Emparejar los individuos de la nueva población, haciendo que intercambien material genético, y que alguno de los bits de un gen se vea alterado debido a una mutación espontánea.

Cada uno de los pasos enunciados consiste en una actuación sobre las cadenas de bits, es decir, la aplicación de un *operador* a una cadena binaria. Se les denominan, por razones obvias, *operadores genéticos,* y hay tres principales: *selección, crossover*

o recombinación y *mutación;* todos estos operadores serán explicados a continuación.

Un algoritmo genético debe también definir una serie de parámetros que se deben definir para su funcionamiento.

1. ***Tamaño de la población:*** Debe ser suficiente para garantizar la diversidad de las soluciones, y, además, tiene que crecer más o menos con el número de bits del cromosoma, aunque nadie ha aclarado cómo tiene que hacerlo. Por supuesto, depende también del ordenador en el que se esté ejecutando.
2. ***Condición de terminación:*** Lo más habitual es que la condición de terminación sea la convergencia del algoritmo genético o un número prefijado de generaciones.
3. ***Evaluación y selección:*** Durante la evaluación se decodifica el gen, convirtiéndose en una serie de parámetros de un problema, se halla la solución del problema a partir de esos parámetros, y se le da una puntuación a esa solución en función de lo cerca que esté de la mejor solución. A esta puntuación se le llama *fitness*. El *fitness* determina siempre los cromosomas que se van a reproducir, y aquellos que se van a eliminar, pero hay varias formas de considerarlo para seleccionar la población de la siguiente generación:

a. Usar el orden, o rango, y hacer depender la probabilidad de permanencia o evaluación de la posición en el orden.

b. En algunos casos, el fitness no es una sola cantidad, sino diversos números, que tienen diferente consideración. Basta con que tal fitness forme un orden parcial, es decir, que se puedan comparar dos individuos y decir cuál de ellos es mejor. Esto suele suceder cuando se necesitan optimizar varios objetivos.

Una vez evaluado el fitness se tiene que crear la nueva población teniendo en cuenta que los *buenos* rasgos de los mejores se transmitan a esta. Para ello hay que seleccionar una serie de individuos encargados de tan ardua tarea. Y esta selección, y la consiguiente reproducción, se pueden hacer de tres formas principales:

1. *Basado en el rango:* En este esquema se mantiene un porcentaje de la población, generalmente la mayoría, para la siguiente generación. Se coloca toda la población por orden de fitness, y los M menos dignos son eliminados y sustituidos por la descendencia de alguno de los M mejores con algún otro individuo de la población. A este esquema se le pueden aplicar otros criterios; por ejemplo, se crea la descendencia de uno de los paladines/amazonas, y esta sustituye al más parecido entre los perdedores. Esto se denomina *crowding,* y fue introducido por DeJong. También es posible que cuando nazca una nueva criatura se seleccionen k individuos de la población, y se elimina al más parecido a la nueva criatura. Una variante de éste es el muestreado estocástico universal, que trata de evitar que los individuos con más fitness copen la población; en vez de dar la vuelta a una ruleta con una ranura, da la vuelta a la ru-

leta con N ranuras, tantas como la población; de esta forma la distribución estadística de descendientes en la nueva población es más parecida a la real.

2. *Rueda de ruleta:* Se crea un *pool* genético formado por cromosomas de la generación actual, en una cantidad proporcional a su fitness (ver Figura 4). Si la proporción hace que un individuo domine la población, se le aplica alguna operación de escalado. Dentro de este *pool* se cogen parejas aleatorias de cromosomas y se emparejan, sin importar incluso que sean del mismo progenitor (para eso están otros operadores, como la mutación). Hay otras variantes: por ejemplo, en la nueva generación se puede incluir el mejor representante de la generación actual. En este caso se denomina *método elitista.*
3. *Selección de torneo:* Se escoge aleatoriamente un número T de individuos de la población, y el que tiene puntuación mayor se reproduce, sustituyendo su descendencia al que tiene menor puntuación.
4. ***Proceso de crossover:*** Consiste en el intercambio de material genético entre dos cromosomas (a veces más, como el *operador orgía.*). El *crossover* es el principal operador genético, hasta el punto que se puede decir que no es un algoritmo genético si no tiene *crossover,* y, sin embargo, puede serlo perfectamente sin mutación, según descubrió Holland. El *teorema* de los *esquemas* confía en él para hallar la mejor solución a un problema, combinando soluciones parciales. Para aplicar el crossover, entrecruzamiento o recombinación, se escogen aleatoriamente dos miembros de la población. Esta selección puede emparejar incluso a dos descendientes de los mismos padres sin que ello represente en sí un problema, lo que se puede garantizar con este "incesto" es la perpetuación de un individuo con buena puntuación. Por otro lado, si esto sucede demasiado a menudo, puede crear problemas: toda la población puede aparecer dominada por los descendientes de algún gen, que, además, puede tener caracteres no

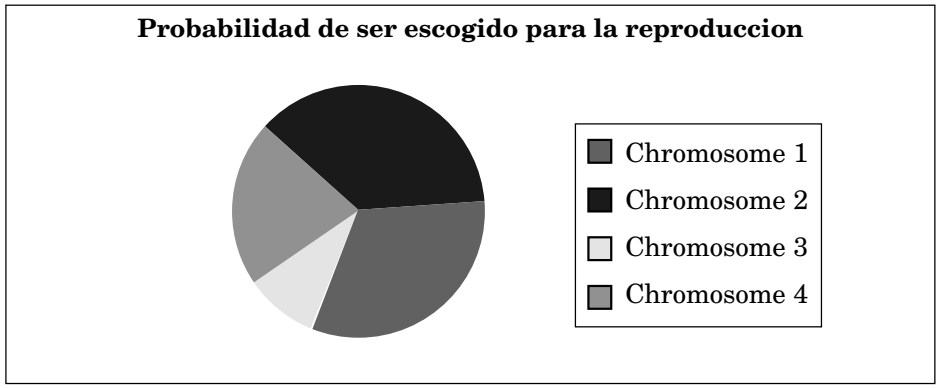

Figura 4 Probabilidad de cada cromosoma de ser escogido para la siguiente generación.

deseados. Lo anterior se suele denominar en otros métodos de optimización como *atranque en un mínimo local,* y es uno de los principales problemas con los que se enfrentan los que aplican algoritmos genéticos.[5]

En cuanto al teorema de los esquemas, se basa en la noción de *bloques de construcción.* Una buena solución a un problema está constituida por unos buenos bloques. El *crossover* es el encargado de mezclar bloques buenos que se encuentren en los diversos progenitores, y que serán los que den a los mismos una buena puntuación. La presión selectiva se encarga de que sólo los buenos bloques se perpetúen, y poco a poco vayan formando una buena solución. *El teorema de los esquemas* viene a decir que la cantidad de *buenos bloques* se va incrementando con el tiempo de ejecución de un algoritmo genético, y es el resultado teórico más importante en algoritmos genéticos.

El intercambio genético se puede llevar a cabo de muchas formas, pero hay dos grupos principales.[6]

1. *Crossover n-puntos:* Los dos cromosomas se cortan por *n* puntos, y el material genético situado entre ellos se intercambia. Lo más habitual es un crossover de un punto o de dos puntos; en las figuras 5 y 6 se ilustra este proceso.

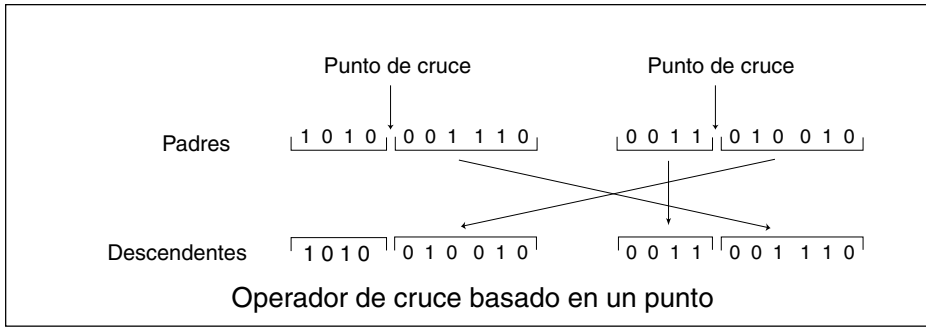

Figura 5. Ejemplo de cruce basado en punto.

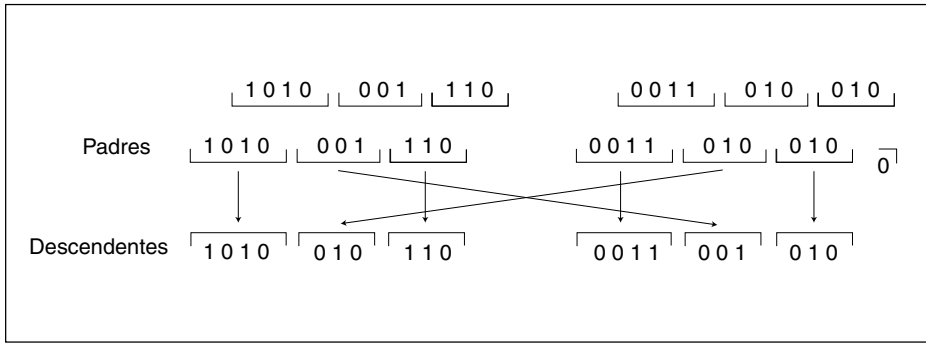

Figura 6. Ejemplo de cruce basado en dos puntos.

2. *Crossover uniforme:* Se genera un patrón aleatorio de 1s y 0s, y se intercambian los bits de los dos cromosomas que coincidan donde hay un 1 en el patrón. O bien, se genera un número aleatorio para cada bit, y si supera una determinada probabilidad se intercambia ese bit entre los dos cromosmas (ver Figura 7).

5. ***Mutación:*** En la Evolución, una mutación es un suceso poco común que sucede aproximadamente en una de cada mil replicaciones, y aunque en la mayoría de los casos las mutaciones son letales, en general contribuyen a la diversidad genética de la especie. En un algoritmo genético tendrán el mismo papel y la misma baja frecuencia. Una vez establecida la frecuencia de mutación, por ejemplo, uno por mil, se examina cada bit de cada cadena cuando se vaya a crear la nueva criatura a partir de sus padres (normalmente se hace de forma simultánea al crossover). Si un número generado aleatoriamente está por debajo de esa probabilidad, se cambiará el bit (es decir, de 0 a 1 ó de 1 a 0). Si no, se dejará como está. Dependiendo del número de individuos que haya y del número de bits por individuo, puede resultar que las mutaciones sean extremadamente raras en una sola generación.

2.6 Resumen de un algoritmo genético

A modo de resumen se enumerarán los pasos básicos necesarios para la implementación de un algoritmo genético.

1. Obtener la población inicial con la que se va a trabajar.
2. Seleccionar los individuos que serán padres de la siguiente generación. Para esto debe existir una forma de establecer qué tan buenos son los individuos y también debe definirse una estrategia para escoger a estos individuos.
3. Producir hijos a partir de los padres seleccionados mediante el proceso de cruce.

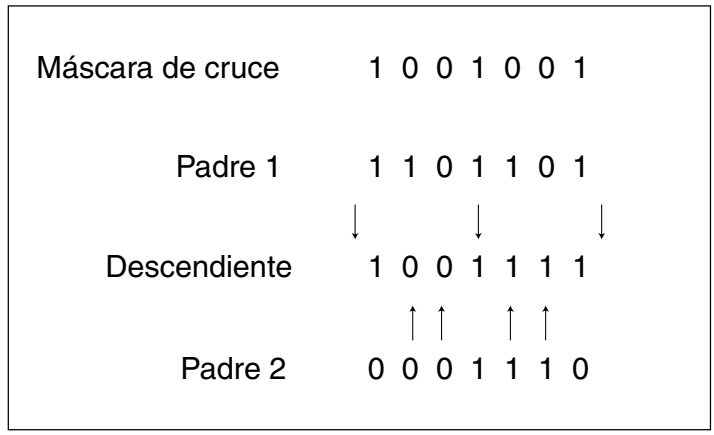

Figura 7. Ejemplo de crossover uniforme.

4. Mutar a algunos de los individuos hijos.
5. Seleccionar los individuos entre padres e hijos que van a pasar a la siguiente generación.
6. Realizar los procesos 2 - 5 hasta que se ha llegado al número de generaciones deseadas, en donde cada ejecución de estos pasos cuenta como una generación.

3. APLICACIÓN DE LOS ALGORITMOS GENÉTICOS AL ALINEAMIENTO DE MÚLTIPLES SECUENCIAS DE AMINOÁCIDOS

La idea básica de este método es intentar generar varios AMS mediante reacomodaciones que simulan la inserción de gaps (huecos) y acciones de recombinación durante la replicación para obtener puntajes más altos para el AMS. Dada la forma en que opera este algoritmo no se garantiza que el resultado final sea el óptimo o el más alto que se pueda alcanzar.[1]

El éxito del algoritmo genético parece radicar en los pasos que se toman en el reacomodamiento de las secuencias, muchos de los cuales simularían cambios ocurridos en el proceso evolutivo de la familia de proteínas. Aunque este algoritmo puede generar AMS para muchas secuencias, el programa es lento para más de veinte cadenas. A continuación se explicarán los pasos del algoritmo, que está basado en la propuesta expuesta en:[1]

1. Las secuencias son escritas en filas, como en una página, excepto que ellas son acomodadas en cadenas que miden un 25% más que la longitud original de la secuencia. La posición a partir de la que es plasmada la secuencia es aleatoria y los espacios se llenan con gaps (-) de modo que los extremos de las nuevas cadenas empaten. Normalmente se crea una población de 100 de estos alineamientos y lo que obtenemos son los 100 primeros posibles alineamientos generados por el algoritmo. En las figuras 8 y 9 podemos ver un ejemplo de cómo se parte de un grupo de secuencias iniciales y se realiza un proceso de reacomodación en una tabla más grande.
2. Cada alineamiento es calificado usando el método de la suma de

x	x	x	x	x	x	x	x
y	y	y	y	y	y	y	y
z	z	z	z	z	z	z	z

Figura 8. Secuencias iniciales.

-	x	x	x	x	x	x	x	x	-
y	y	y	y	y	y	y	y	-	-
-	-	z	z	z	z	z	z	z	z

Figura 9. Secuencias reacomodadas.

pares (se tienen en cuenta todas las posibles combinaciones de aminoácidos en cada columna del alineamiento) y tomando las puntuaciones de la matriz Blosum.

3. Se esogen los 50 mejores alineamientos y pasan a la segunda generación sin cambios, a continuación se realiza un sorteo para determinar cuáles serán los 50 alineamientos restantes, siguiendo un proceso similar a sacar canicas de una bolsa y en el que mientras mayor sea la puntuación del AMS mayor es la probabilidad de ser escogido. La segunda mitad pasará por la fase de mutación que se explica a continuación.
4. En el proceso de mutación la secuencia no debe ser modificada en las letras que la representan, porque de lo contrario no sería un alineamiento, pero sí son insertados gaps y son reacomodadas las cadenas en un intento de obtener mejores puntajes en el AMS. Aunque existen varias maneras de manejar la inserción de gaps, en nuestra versión del algoritmo la inserción se hará de manera aleatoria tanto en lo que respecta a la posición de la cadena como a la longitud del gap. En las figuras 10 y 11 se presenta un conjunto de secuencias alineadas antes y después de la inserción de gaps en el alineamiento respectivamente.
5. La fase de recombinación se lleva a cabo de la siguiente forma. A partir de la totalidad de alineamientos se seleccionan padres usando el método de selección por torneo en el que son escogidos dos alineamientos al azar y el que tenga la mejor puntuación será uno de los padres; el proceso se repite para obtener el segundo alineamiento padre.

Aleatoriamente se genera un número que nos va a indicar el punto de corte del primer alineamiento, que es el número del carácter por el que vamos a realizar el primer corte, como se muestra en las figuras 12 y 13.

-	x	x	x	x	x	x	x	x	-
y	y	y	y	y	y	y	y	-	-
-	-	z	z	z	z	z	z	z	z

Figura 10. Alineamiento con las secuencias reacomodadas.

-	x	x	x	-	x	x	-	x	x	x	-	-
y	y	y	-	y	y	-	y	y	y	-	-	-
-	-	z	-	-	z	z	z	-	z	z	z	z

Figura 11. Alineamiento tras la inserción de gaps en las secuencias.

N=5

xx-xx-Axx-xxx
-xxx-x-A-x-xx-xx
x-x-xx-Axx-x-x-x
x-xxx-A-x-xxxx

Figura 12. Padre 1.

-xxx-x-A-x-xx-xx
x-x-xx-Axx-x-x-x
xx-xx-Axx-xxx
x-xxx-x-A-xxxx

Figura 13. Padre 2.

Lo que sigue es pegar las partes sombreadas entre sí obteniendo así un nuevo alineamiento que sería el hijo 1; de igual forma se pegarían las partes no sombreadas creando un hijo 2. Se escoge naturalmente al alineamiento hijo con mejor puntaje. Por otro lado se pueden buscar escoger los mejores *n* alineamientos y pasarlos intactos a la siguiente generación o, por el contrario, permitir que todos los alineamientos de la nueva generación sean producto de la reproducción de dos AMS.

6. A continuación se parte de la nueva población, como si fuera la original, y la llevamos de nuevo al paso dos. Este proceso se realiza generalmente 100 veces, pero puede llegar a ser ejecutado incluso tanto como 1.000 veces.
7. El proceso completo de producir un conjunto de AMS mediante la replicación y mutación es repetido muchas veces para obtener, así mismo, un gran número de posibles AMS y es escogido el mejor calificado.

4. EJEMPLO DE LA APLICACIÓN DEL ALGORITMO

A continuación veremos un ejemplo de cómo funciona el algoritmo paso a paso siguiendo el alineamiento de tres secuencias.

En primer lugar hay que resaltar que el programa permite parametrizar algunos valores necesarios para el alineamiento. Estos valores son: el tamaño de la población, el número de generaciones que se van a producir, el número de gaps que se van a insertar en cada cadena en el proceso de mutación y el tamaño máximo de gap que será insertado en la mutación. Para el ejemplo solo se va a trabajar con tres cadenas de una longitud relativamente corta, se generarán la primera y parte de la segunda generación, y el número de gaps a insertar por mutación será de uno igual al del tamaño máximo del gap.

1. En la Figura 14 se muestran las secuencias cn las que se va a trabajar:

A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T
N	D	C	Q	D	F	F	F	T	Q	I	K	T			
H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F

Figura 14. Secuencias iniciales.

2. Se calcula el 25% de la longitud de la cadena más larga y eso nos da los espacios extras que se van a usar para mover las cadenas a un lado y al otro; en este caso la holgura es de cuatro y a continuación se genera un número aleatorio entre 0 y 4 para cada cadena y eso determina el desplazamiento de la misma. Con los parámetros anteriores se generan cuatro posibles alineamientos, estos son mostrados en la Figura 15.
3. A continuación se saca el puntaje de cada alineamiento teniendo en cuenta las calificaciones establecidas en la matriz Blosum62; se ordenan y se escoge la primera mitad que corresponde a la de mejores calificaciones y pasa intacta. Se pueden ver en la Figura 16 los alineamientos con mayor número de coincidencias en este caso,
4. A continuación la otra mitad de los alineamientos pasa por un proceso de mutación en el que para cada secuencia se escoge aleatoriamente el número, el tamaño y la posición en la que va a ser insertado cada uno de los gaps, teniendo en cuenta, claro está, los parámetros introducidos al programa. La selección de esta segunda mitad se lleva a cabo así:

- Primero se establece un rango que va de la menor a la mayor puntuación de entre todos los alineamientos.

-	-	A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-
-	-	-	N	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-
-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-	-

A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-	-	-
-	N	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-	-	-
-	-	-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-

-	-	-	-	A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T
-	-	N	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-	-
-	-	-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-

-	A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-	-
N	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-	-	-	-
-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-	-

Figura 15. Alineamientos inicales.

-	-	A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-
-	-	-	N	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-
-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-	-

A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-	-	-
-	N	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-	-	-
-	-	-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-

Figura 16. Alineamientos con mayor puntuación.

- Luego se generan números aleatorios dentro de ese rango y se escogen posiciones aleatorias de entre el conjunto de alineamientos, se compara el puntaje aleatorio con el del alineamiento aleatorio; si el puntaje del alineamiento es mayor o igual, es seleccionado, de lo contrario se repite el proceso. Con el proceso anterior se busca darle una mayor probabilidad a los alineamientos "fuertes" de ser elegidos. La Figura 17 muestra el resultado del proceso anterior.

5. Tanto la primera mitad de los mejores como la segunda mitad de los alineamientos mutados se juntan y luego son calificados y reordenados (ver Figura 18).

A	R	N	D	-	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-	-
-	N	D	C	Q	D	F	F	-	F	T	Q	I	K	T	-	-	-	-	-
-	-	-	H	I	G	A	Q	R	P	S	T	-	N	D	C	Q	E	F	F

-	A	R	N	D	C	Q	F	F	F	T	-	Q	I	L	K	S	T	-	-
N	-	D	C	Q	D	F	-	F	F	T	Q	I	K	T	-	-	-	-	-
-	H	I	G	A	-	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-

Figura 17. Alineamientos tras la inserción de gaps.

-	-	A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-
-	-	-	N	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-
-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-	-

-	A	R	N	D	C	Q	F	F	F	T	-	Q	I	L	K	S	T	-	-
N	-	D	C	Q	D	F	-	F	F	T	Q	I	K	T	-	-	-	-	-
-	H	I	G	A	-	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-

A	R	N	D	-	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-	-
-	N	D	C	Q	D	F	F	-	F	T	Q	I	K	T	-	-	-	-	-
-	-	-	H	I	G	A	Q	R	P	S	T	-	N	D	C	Q	E	F	F

-	-	-	-	A	R	N	D	C	-	Q	F	F	F	T	Q	I	L	K	S	T
-	-	N	D	-	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-	-
-	-	-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-

Figura 18. Alineamientos seleccionados y reordenados.

6. Ahora se realiza el proceso de reproducción basado en el método de torneo en el cual selecciono aleatoriamente una pareja de alineamientos; aquí el que tenga una mejor calificación va a ser escogido como padre. Este proceso se repite n veces (donde n es el tamaño de la población), ya que cada iteración va a asegurar obtener un alineamiento para agregar a la nueva población. A continuación se va a ver en detalle cómo es el proceso de reproducción. Para tal propósito se va a trabajar con los dos primeros alineamientos de la Figura 19.
7. A partir de estos dos alineamientos se genera un número aleatorio dentro del rango de cero hasta la longitud de la cadena más corta, ese será el punto de corte. Para el ejemplo el número será 5, esto indica no la posición sino el carácter por el que se realizará el corte (ver Figura 20).
8. A continuación se intercambia la primera parte de la matriz superior con la segunda de la inferior y viceversa y se obtiene algo como lo que muestra la Figura 21 c.
9. A continuación se escoge al hijo que tenga la mejor calificación y es agregado a la nueva generación. El proceso se muestra en la Figura 22
10. Cabe anotar que en el proceso de reproducción se eliminan las columnas llenas de gaps que se en-

-	-	A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-
-	-	-	N	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-
-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-	-

-	A	R	N	D	C	Q	F	F	F	T	-	Q	I	L	K	S	T	-	-
N	-	D	C	Q	D	F	-	F	F	T	Q	I	K	T	-	-	-	-	-
-	H	I	G	A	-	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-

Figura 19. Alineamientos escogidos para el proceso de cruce.

-	-	A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-
-	-	-	N	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-
-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-	-

-	A	R	N	D	C	Q	F	F	F	T	-	Q	I	L	K	S	T	-	-
N	-	D	C	Q	D	F	-	F	F	T	Q	I	K	T	-	-	-	-	-
-	H	I	G	A	-	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-

Figura 20. Identificación del carácter de corte.

-	-	A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-
-	-	-	N	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-
-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-	-

a. Padre 1.

-	A	R	N	D	C	Q	F	F	F	T	-	Q	I	L	K	S	T	-	-
N	-	D	C	Q	D	F	-	F	F	T	Q	I	K	T	-	-	-	-	-
-	H	I	G	A	-	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-

b. Padre 2.

-	-	A	R	N	D	C	Q	F	F	F	T	-	Q	I	L	K	S	T
-	-	-	N	D	C	Q	D	F	-	F	F	T	Q	I	K	T	-	-
-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-

-	A	R	N	D	C	Q	F	F	F	T	Q	I	L	K	S	T	-	
N	-	D	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-	
-	H	I	G	A	-	Q	R	P	S	T	N	D	C	Q	E	F	F	

Figura 21c. Hijo 1 e hijo 2. Las partes sombreadas corresponden al padre uno, las otras al padre dos.

-	-	A	R	N	D	C	Q	F	F	F	T	-	Q	I	L	K	S	T
-	-	-	N	D	C	Q	D	F	-	F	F	T	Q	I	K	T	-	-
-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-

Hijo del alineamiento uno y dos cortado por el quinto carácter.

A	R	N	D	-	C	Q	F	F	F	T	Q	I	L	K	S	T	-	-
-	N	D	C	Q	D	F	F	-	F	T	Q	I	K	T	-	-	-	-
-	-	-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F

Hijo del tercer y cuarto alineamiento, cortados por el octavo carácter.

A	R	N	D	-	C	Q	F	F	F	T	-	Q	I	L	K	S	T	-	-	-	-	
-	N	D	C	Q	D	F	F	-	F	T	Q	I	K	T	-	-	-	-	-	-	-	
-	-	-	H	I	G	A	Q	R	P	S	T	-	N	T	N	D	C	Q	E	F	F	

Hijo del segundo y tercer alineamiento cortados por el décimo carácter.

-	-	A	R	N	D	C	-	Q	F	F	F	T	Q	I	L	K	S	T
N	D	-	C	Q	D	F	F	F	T	Q	I	K	T	-	-	-	-	-
-	H	I	G	A	Q	R	P	S	T	N	D	C	Q	E	F	F	-	-

Figura 22. Escogencia del hijo con menor calificación.

cuentran al final de los alineamientos. Por otro lado, cuando se han generado el número de hijos suficientes para cumplir con el total de la población, se retorna al paso tres para seguir con el procedimiento hasta completar el número solicitado de generaciones. Es importante anotar que para este ejemplo solo se está trabajando con tres secuencias cortas, una población de cuatro y tan solo dos generaciones. La idea es que se trabaje con no más de 20 secuencias, con una población de 100 individuos y 100 generaciones.

5. CONCLUSIONES

Tras haber realizado algunas pruebas con diversos grupos de secuencias, se evidenciaron varios aspectos que hay que tener en cuenta para obtener buenos resultados con este algoritmo. Dado que uno de los problemas de los algoritmos genéticos en general es la existencia de los máximos locales, puede ser necesario aplicar el algoritmo varias veces sobre el mismo conjunto de secuencias y escoger entre los diferentes resultados el mejor. Por otro lado, es necesario jugar un poco con los parámetros del algoritmo tales como el número y longitud de gaps que se pueden insertar en las secuencias mutadas y el tamaño de la población. Esta variación en los parámetros es muy importante porque no es lo mismo trabajar con secuencias que son muy similares entre sí a trabajar con secuencias ampliamente dispares.

Con las pruebas también se pudo detectar que cuando se está realizando la implementación del algoritmo, es de vital importancia evitar que las secuencias aumenten de tamaño de manera exagerada porque si no se toman tales precauciones, puede suceder que todos los alineamientos estén llenos de secuencias que son en un alto porcentaje gaps; y si este error se propaga, el resultado del algoritmo puede ser muy pobre. Por último, es de vital importancia ejecutar este algoritmo en una máquina relativamente robusta, especialmente en la medida en que aumentamos el número y la longitud de las cadenas que vamos a alinear.

A lo largo de la elaboración de la aplicación se pudo observar una característica propia de los algoritmos genéticos y es la gran flexibilidad que brindan. Aunque es cierto que en un principio lo que se busca es representar el problema en términos de cromosomas codificados que evidencian parte de la rigidez del modelo, a medida que se exploran las opciones de desarrollo del algoritmo se ve claramente lo que se quiere expresar. El hecho que el algoritmo propuesto busque lograr que toda la población de cromosomas tenga puntajes altos o por el contrario el enfoque en el que se busca solamente un gran máximo va más allá del simple significado matemático. Lo que se tiene en frente es un sistema que nos puede mostrar diferentes soluciones a un problema y es ahí donde la capacidad de interpretar esas propuestas puede hacer la diferencia.

6. BIBLIOGRAFÍA

1. Mount, David W. *Bioinformatics: sequence and genome analysis,* Cold Spring Harbor, 2001.

2. Lesk, Arthur M. *Introduction to bioinformatics,* Oxford University Press, 2002.
3. Attwood, Teresa K., Parry-Smith, David J. *Introducción a la bioinformática,* Prentice Hall, 2002, Madrid.
4. Goldberg, D. E. *Genetic algorithms in search, optimization, and machine learning.* Reading, MA: Addison-Wesley. 1989.
5. http://geneura.ugr.es/~jmerelo/ie/ags.htm
6. http://www.sc.ehu.es/isg/

CURRÍCULO

Luis Felipe Solanilla: Bachiller del Colegio Lacordaire. Estudiante de décimo semestre de Ingeniería de Sistemas de la Universidad ICESI. Trabajó como monitor durante cuatro años en el grupo Help Desk de la Universidad ICESI.

Luis Eduardo Múnera: Matemático de la Universidad del Valle. Máster y Doctor en Informática de la Universidad Politécnica de Madrid. Docente-Investigador de la Universidad ICESI.